Lab 5

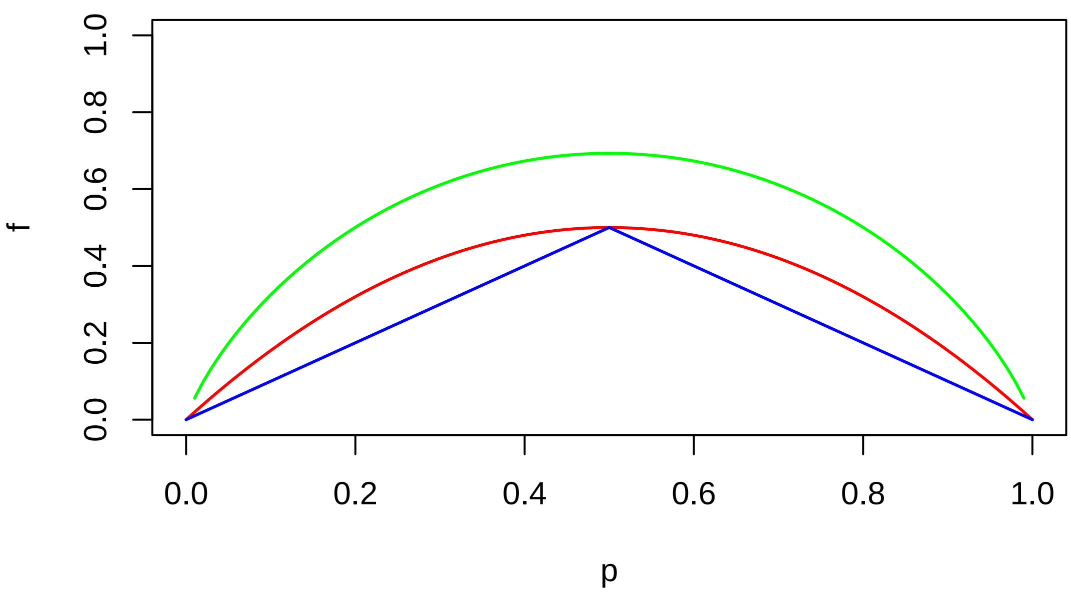
Zara Waheed

March 12, 2022

Question 8.3

```
p <- seq(0,1,0.01)

#For two classes
gini <- 2*p*(1-p)
err <- 1 - pmax(p, 1-p)
ent <- -(p*log(p) + (1-p)*log(1-p))
plot(NA, xlim = c(0,1), ylim = c(0,1), xlab = "p", ylab = "f")
lines(p, gini, type = "l", col = "red", lwd = 1.5)
lines(p, err, type = "l", col = "blue", lwd = 1.5)
lines(p, ent, type = "l", col = "green", lwd = 1.5)
```



Question 8.5

Majority vote:

P is greater than 0.5 6/10 times so final classification is Red.

Average probability:

10 estimates = 0.45 each P(Class is Red | X) < 0.5, so final classification is Green.

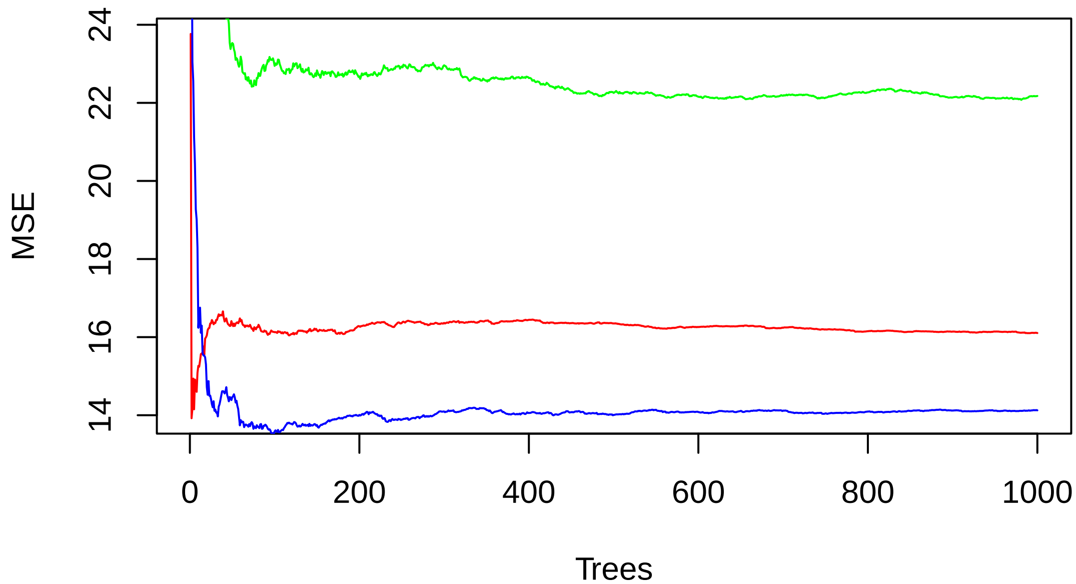
Question 8.7

```
data("Boston")
set.seed(100)

train <- sample(1:nrow(Boston), nrow(Boston)/2)
Boston_train <- Boston[train, -14]
Boston_test <- Boston[-train, -14]
y_train <- Boston[train, 14]
y_test <- Boston[-train, 14]

fit1 <- randomForest(Boston_train, y = y_train, xtest = Boston_test, ytest = y_test, mtry = ncol(Boston
fit2 <- randomForest(Boston_train, y = y_train, xtest = Boston_test, ytest = y_test, mtry = (ncol(Bosto
fit3 <- randomForest(Boston_train, y = y_train, xtest = Boston_test, ytest = y_test, mtry = sqrt(ncol(Bo

plot(1:1000, fit1$test$mse, type = "l", col = "red", xlab = "Trees", ylab = "MSE")
lines(1:1000, fit2$test$mse, type = "l", col = "blue")
lines(1:1000, fit3$test$mse, type = "l", col = "green")
```



Question 8.8

a)

```
data("Carseats")
set.seed(100)
s <- sample(1:nrow(Carseats), nrow(Carseats)*0.7)
cs_train <- Carseats[s, ]
cs_test <- Carseats[-s, ]
```

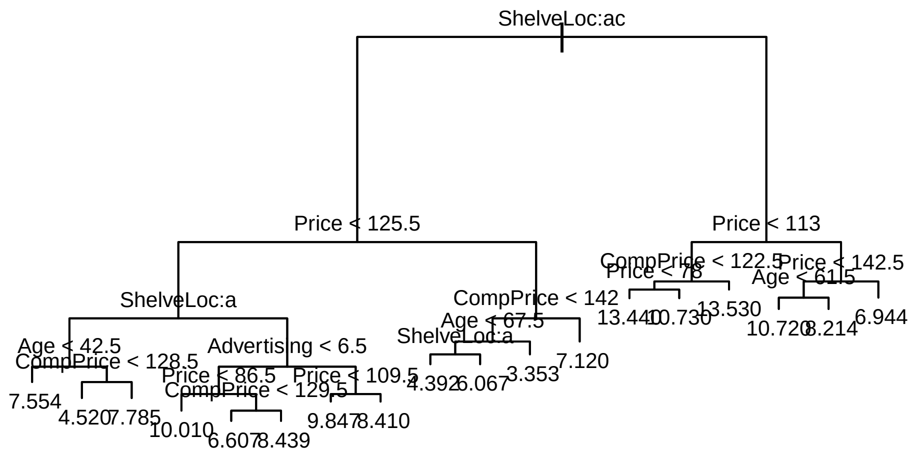
b)

```
rt <- tree(Sales ~ ., data = cs_train)
summary(rt)
```

```
##
```

```
## Regression tree:
## tree(formula = Sales ~ ., data = cs_train)
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Price"       "Age"         "CompPrice"   "Advertising"
## Number of terminal nodes:  18
## Residual mean deviance:  2.246 = 588.6 / 262
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -4.10700 -1.01200  0.04947  0.00000  0.86690  4.03300
```

```
plot(rt)
text(rt, cex = 0.65)
```

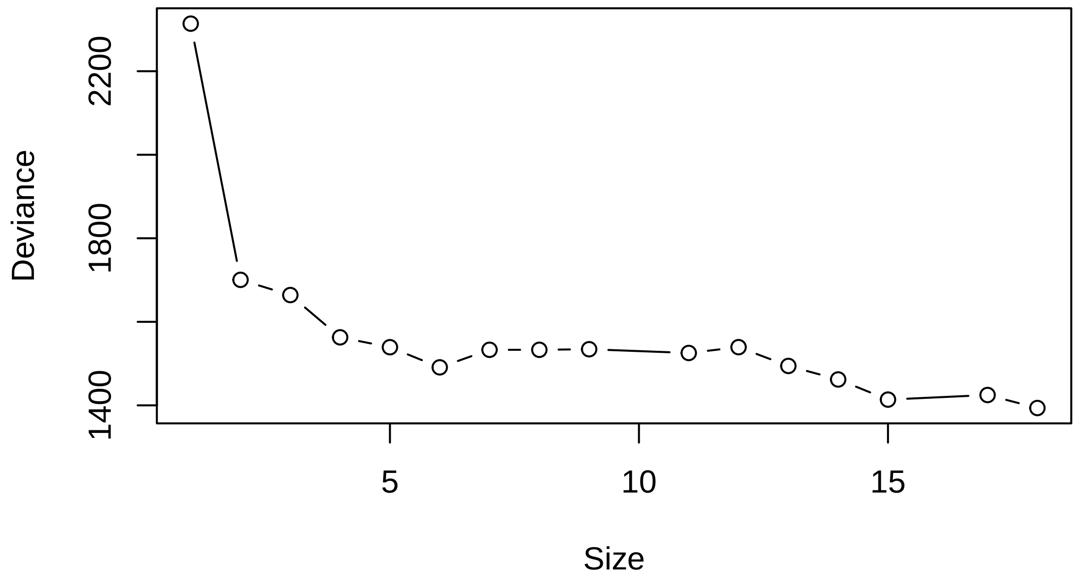



```
pred_rt <- predict(rt, cs_test)
mse_rt <- mean((cs_test$Sales - pred_rt)^2)
mse_rt
```

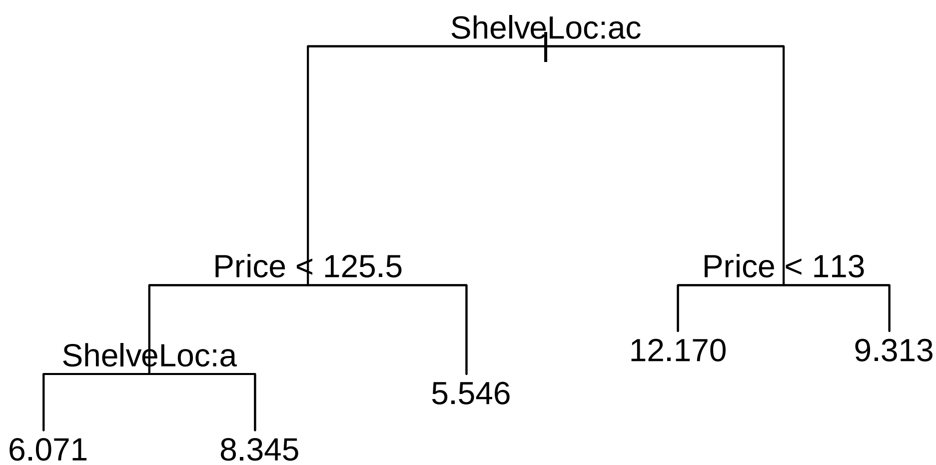
```
## [1] 5.545939
```

c)

```
cv_rt <- cv.tree(rt)
plot(cv_rt$size, cv_rt$dev, xlab = "Size", ylab = "Deviance", type = "b")
```




```
# Pruning
prune_rt <- prune.tree(rt, best = 5)
plot(prune_rt)
text(prune_rt)
```




```
prune_pred <- predict(prune_rt, cs_test)
prune_mse <- mean((prune_pred - cs_test$Sales)^2)
prune_mse
```

```
## [1] 5.898366
```

The pruned tree gives a higher MSE than the unpruned tree so it is not improving the results.

d)

```
bagging <- randomForest(Sales ~ ., data = cs_train, mtry = 10, importance = TRUE, ntree = 500)
bagging_pred <- predict(bagging, cs_test)
bagging_mse <- mean((bagging_pred - cs_test$Sales)^2)
bagging_mse
```

```
## [1] 2.674554
```

Bagging lowers the MSE

```
importance(bagging)
```

```
##                %IncMSE IncNodePurity
## CompPrice   32.2423849    211.128709
## Income       5.9048764    106.908832
## Advertising 23.4342438    164.196900
## Population   2.2913546     80.505455
## Price       66.6550819    601.865374
## ShelveLoc   78.8806860    804.518153
## Age         18.7810422    180.865214
## Education    0.5069418     58.727420
## Urban       -0.2938359      7.946885
## US           6.3341555     14.222222
```

ShelveLoc, Price and Advertising seem to rank highest in importance

e)

```
rf_mse <- c()
for (i in 1:10) {
  rf <- randomForest(Sales ~ ., data = cs_train, mtry = i, importance = TRUE, ntree = 500)
  rf_pred <- predict(rf, cs_test)
  rf_mse[i] <- mean((rf_pred - cs_test$Sales)^2)
}

which.min(rf_mse)
```

```
## [1] 6
```

```
rf_mse[which.min(rf_mse)]
```

```
## [1] 2.651789
```

9 variables MSE seems slower than bagging and trees

```
importance(rf)
```

```
##               %IncMSE IncNodePurity
## CompPrice   32.405529    213.562271
## Income       5.788450    108.021614
## Advertising 22.491350    160.393239
## Population  -0.702572     77.723427
## Price       65.532883    607.324374
## ShelveLoc   77.480706    817.276834
## Age         16.338186    181.865983
## Education    2.221488     58.173260
## Urban       -2.170568      7.918909
## US           6.820931     13.961101
```

ShelveLoc,Price and CompPrice seem to rank highest in importance

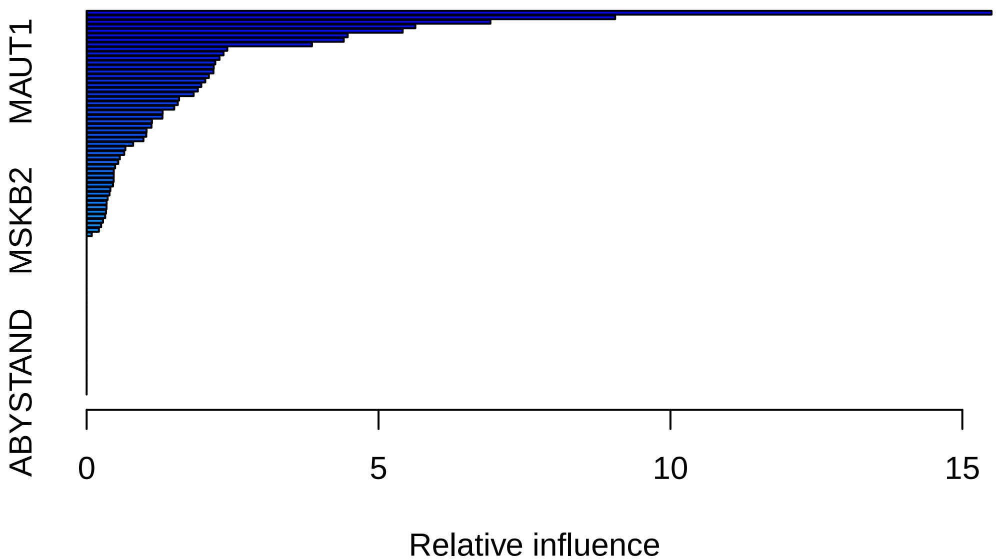
Question 8.11

a)

```
data("Caravan")
Caravan$Purchase <- ifelse(Caravan$Purchase == "No", 0, 1)
caravan_train <- Caravan[1:1000, ]
caravan_test <- Caravan[1001:5822, ]
```

b)

```
set.seed(100)
fit_boost <- gbm(Purchase ~ ., data = caravan_train, shrinkage = 0.01, n.trees = 1000, distribution = "
kable(summary(fit_boost), row.names = F)
```



var	rel.inf
PPERSAUT	15.4998902
MKOOPKLA	9.0528949
MOPLHOOG	6.9160655
PBRAND	5.6302222
MBERMIDD	5.4132527
MINK3045	4.4711506
MGODGE	4.4040679
ABRAND	3.8593845
MSKA	2.4089808
MSKC	2.3448713
MAUT2	2.2758250
PWAPART	2.2035400
MBERARBG	2.1765534
MAUT1	2.1715923
MOSTYPE	2.0935338
MGODPR	2.0318846
MFWEKIND	1.9637348
MINKGEM	1.9044302
MRELGE	1.8328305
MAUT0	1.5832990
MGODOV	1.5607274

var	rel.inf
PBYSTAND	1.5001749
MBERHOOG	1.3015867
MSKB1	1.2974955
MRELOV	1.1183356
MFGEKIND	1.1099018
MINK7512	1.0264714
MHKOOP	1.0215165
MGODRK	0.9725466
MINKM30	0.7960163
MHHUUR	0.6628492
MOPLMIDD	0.6433588
MBERBOER	0.5682018
PLEVEN	0.5402277
MINK4575	0.4876859
MGEMOMV	0.4643345
MSKD	0.4637065
MGEMLEEF	0.4627861
MFALLEEN	0.4513166
PMOTSCO	0.4061700
MZPART	0.3913365
MZFONDS	0.3591340
MBERARBO	0.3415115
APERSAUT	0.3403718
MOSHOOFD	0.3313472
MINK123M	0.3180736
MSKB2	0.2802097
MRELSA	0.2481110
MOPLLAAG	0.2090313
MBERZELF	0.0874592
MAANTHUI	0.0000000
PWABEDR	0.0000000
PWALAND	0.0000000
PBESAUT	0.0000000
PVRAAUT	0.0000000
PAANHANG	0.0000000
PTRACTOR	0.0000000
PWERKT	0.0000000
PBROM	0.0000000
PPERSONG	0.0000000
PGEZONG	0.0000000
PWAOREG	0.0000000
PZEILPL	0.0000000
PPLEZIER	0.0000000
PFIETS	0.0000000
PINBOED	0.0000000
AWAPART	0.0000000
AWABEDR	0.0000000
AWALAND	0.0000000
ABESAUT	0.0000000
AMOTSCO	0.0000000
AVRAAUT	0.0000000
AAANHANG	0.0000000

var	rel.inf
ATRACTOR	0.0000000
AWERKT	0.0000000
ABROM	0.0000000
ALEVEN	0.0000000
APERSONG	0.0000000
AGEZONG	0.0000000
AWAOREG	0.0000000
AZEILPL	0.0000000
APLEZIER	0.0000000
AFIETS	0.0000000
AINBOED	0.0000000
ABYSTAND	0.0000000

PPERSAUT, MKOOPKLA, MOPLHOOG, PBRAND and MBERMIDD seem highest in importance.

c)

```r
# Boosting
pred_boost <- predict(fit_boost, caravan_test, n.trees = 1000, type = "response")
boost <- ifelse(pred_boost > 0.2, 1, 0)
table(caravan_test$Purchase, boost)
```

```
##    boost
##        0    1
##   0 4413  120
##   1  255   34
```

$34/(34 + 255) = 2/17$ people end up making purchases from boosting.

```r
# Logistic Regression
caravan_lr <- glm(Purchase ~ ., data = caravan_train, family = binomial)
pred <- predict(caravan_lr, caravan_test, type = "response")
pred_lr <- ifelse(pred > 0.2, 1, 0)
table(caravan_test$Purchase, pred_lr)
```

```
##    pred_lr
##        0    1
##   0 4183  350
##   1  231   58
```

$58/(58 + 231) \sim 1/5$ people end up making purchases from logistic regression which is a better prediction than boosting.